CS4677 Computer Forensics Live System Collection (cont)

Chris Eagle

Fall '06

Environment Variables

- Things like PATH
- Useful to know what their current state is
  - May have been modified by an attacker
- Command: `env`
  - Not a native windows command - `set`
  - Available with cygwin

Linux Kernel Modules (LKM)

- Linux systems have the ability to dynamically extend kernel capabilities via the LKM mechanism
- List of loaded kernel modules
  - `lsmod`

/proc File System

- Virtual file system
- Interface to kernel data on some flavors of Unix
- Of interest
  - `/proc/<pid>`
  - One directory for each running process

/proc/<pid>

```
-r--r--r--    1 root     root  0 Oct 20 22:12 cmdline
-r--r--r--    1 root     root  0 Oct 20 22:12 cpu
lrwxrwxrwx    1 root     root  0 Oct 20 22:12 cwd -> /
-r--------    1 root     root  0 Oct 20 22:12 environ
lrwxrwxrwx    1 root     root  0 Oct 20 22:12 exe ->
                                    /usr/local/sbin/sshd
dr-x------    2 root     root  0 Oct 20 22:12 fd
-r--r--r--    1 root     root  0 Oct 20 22:12 maps
-rw-------    1 root     root  0 Oct 20 22:12 mem
-r--r--r--    1 root     root  0 Oct 20 22:12 mounts
lrwxrwxrwx    1 root     root  0 Oct 20 22:12 root -> /
-r--r--r--    1 root     root  0 Oct 20 22:12 stat
-r--r--r--    1 root     root  0 Oct 20 22:12 statm
-r--r--r--    1 root     root  0 Oct 20 22:12 status
```

/proc/<pid>/exe

- A symbolic link to the program binary
- May have been deleted after the program started
  - Hides the executable from `ls`
- Recover the binary
  - `cat /proc/<pid>/exe > recovered`

System Information

- System Type
  - Windows
    - `systeminfo`
    - `psinfo`
      - http://www.sysinternals.com/ntw2k/freeware/pstools.shtml
  - Unix
    - `uname -a`
    - `uptime`
      - Also available for Windows under cygwin
    - `mount`

MAC Time Summaries

- Useful to obtain prior to performing any forensics data collection
  - Your work will change some of these times
  - Windows
    - `dir /t:a /a /s /o:d c:\`
    - `dir /t:w /a /s /o:d c:\`
    - `dir /t:c /a /s /o:d c:\`
  - Unix
    - `ls -alRu /`
    - `ls -alR /`
    - `ls -alRc /`

Powering Down

- Active debate: clean shutdown vs. pulling the plug
- Clean Shutdown
  - Advantage – maintains file system integrity
  - Disadvantages
    - many changes to file system
    - Malicious process could delete evidence on shutdown
- Pulling power – do it at the wall, not the switch
  - Advantage – maintains state as near running state as possible
  - Disadvantage – possible file system corruption

CS4677 Computer Forensics Network Evidence Collection

Chris Eagle

Fall '06

Classes of Data

- Full Content
  - Every byte of every packet
- Session Data
  - End point information
  - Who talked to who, when and how
- Alert Data
  - Triggered based on predefined event criteria
- Statistical Data
  - Averages of type, time, quantity, etc

Full Content Data

- When to collect?
  - Pre-incident
    - Requires proper planning
    - Large storage requirement
    - Guaranteed coverage of entire event
      - Unless encryption in use
  - Post-incident
    - Intent is to observe incident related activity

Full Content Collection

- Hardware
  - Hubs
    - Traffic passed to all ports
    - Performance penalty
  - TAPS
    - Dedicated copying of packets
  - Inline device
    - Bridging devices
  - SPAN ports
    - Specialized ports on commercial switches

tcpdump

- Open source packet sniffing tool
  - http://www.tcpdump.org
  - Same people offer libpcap
    - Packet capture library
  - On Windows you need WinDump/WinPcap
    - http://www.winpcap.org/
- By default, prints a short summary of each received packet

tcpdump Usage

- Dumps output to console (stdout)
- For session data
  - `tcpdump -i eth0`
  - `-i <interface>`
- Output numeric ip/port values with –nn switch
  - `tcpdump -nn -i eth0`
- Change snaplen (amount of packet captured) with –s
  - `-s 1514` max size of an Ethernet frame
  - `-s 0` Causes capture of entire packet

tcpdump Usage (ii)

- Full content monitoring
  - Set snaplen to grab entire packet
  - Write packets to a capture file
    - -w <filename>

```
tcpdump -s 0 -i eth0 -w dump_10_28_03.cap
```

  - No need for –nn switch as raw packet binary data is being recorded

tcpdump Usage (iii)

- tcpdump understands command line packet filtering rules
  - Used to build *Berkeley Packet Filter* (bpf) rules
- Use rules to restrict content to traffic of interest for example
  - host 131.120.14.2
- Remember you will need to sort through all of the collected data

Live Content Monitoring/Analysis

- tcpdump can only save packets to a file or print brief summaries
- If you need to monitor content in real time you will need a tool that can display entire packets in real time
  - Ethereal is the open source tool of choice here
  - We will cover Ethereal when we cover data analysis

tcpdump For Headers

- By default tcpdump only examines the first 68 bytes of a packet
  - The amount of data actually grabbed is called the *snaplen*
  - 68 byte is sufficient to grab
    - Ethernet header – 14 bytes
    - IP header – 20 bytes
    - TCP header – 20 bytes
    - 14 extra bytes which may be tcp/ip options or application layer data

Monitoring Considerations

- Monitoring machine should be able to see desired traffic
  - Consider network architecture
- Monitoring machine should be invisible to the network
  - Can't be seen by other users
  - Has no IP or the null IP 0.0.0.0
  - Consider cutting transmit wire in network cable

Session Data

- Book recommends
  - Argus
    - http://www.qosient.com/argus/
  - Tcptrace
    - http://www.tcptrace.org/
  - Tcpflow – session rebuilding
    - http://www.circlemud.org/~jelson/software/tcpflow/
- Also consider
  - Chaosreader – session rebuilding
    - http://users.tpg.com.au/bdgcvb/chaosreader.html
  - Wireshark
    - http://www.wireshark.org

Alert Data

- Generally NIDS (network intrusion detection system) alerts
  - Snort http://www.snort.org
    - Trigger on any part of packet
    - Can also save full content
    - Can rerun old packets
- Familiarize yourself with you NIDS' logging capabilities

Statistical Data

- Tcpdstat
  - http://staff.washington.edu/dittrich/talks/core02/tools/tools.html
  - Summaries of tcpdump capture files
- Wireshark
  - Also generates statistics

CS4677 Computer Forensics Network Data Analysis

Chris Eagle

Fall '06

Wireshark Basics

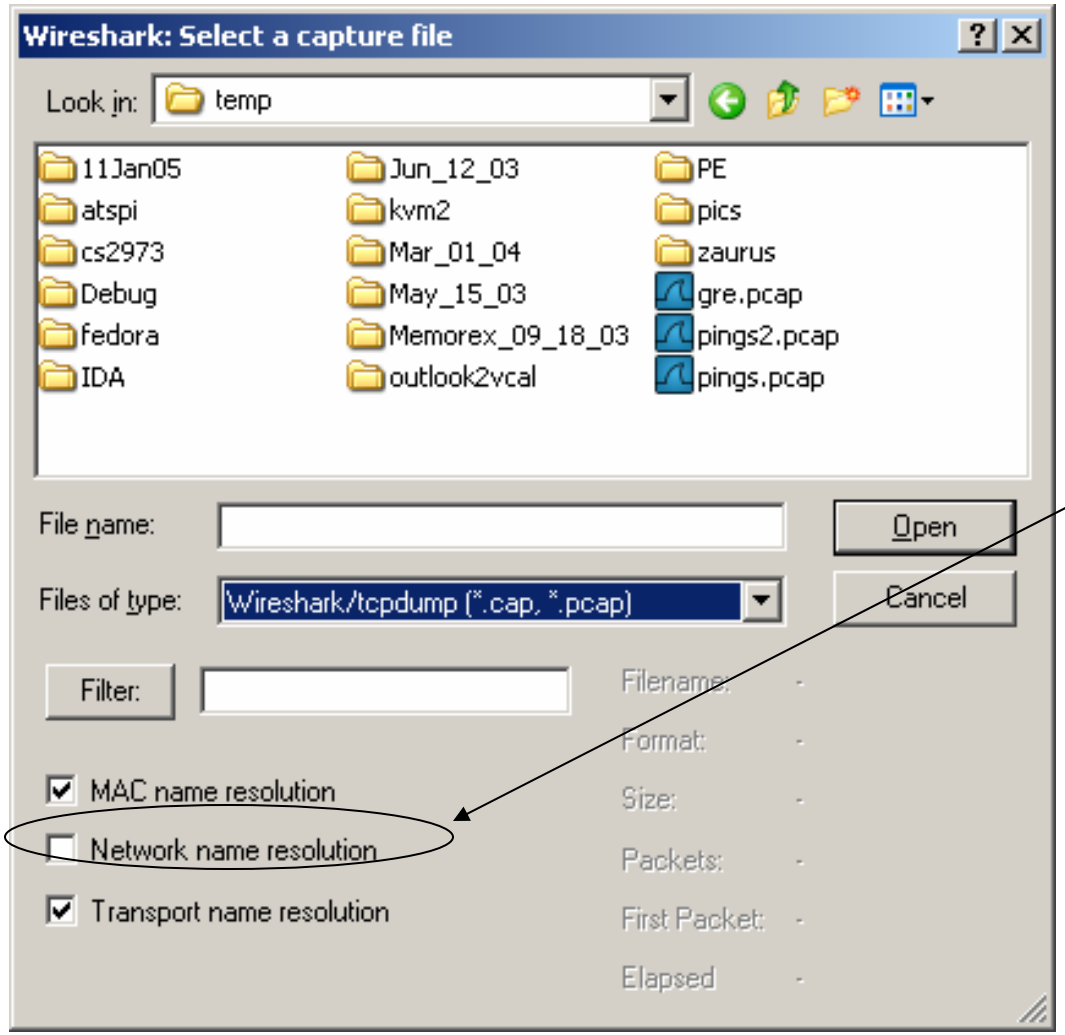
- Live Capture
  - Via Capture/Start menu item
  - Can decode and display packets in real time
  - Capture on interface with no IP or 0.0.0.0 if possible
    - Use a receive only cable if available
- Save to File
  - Via File/Save menu
  - Save captures to file for later analysis
  - If no live analysis is necessary, tcpdump/windump is a better tool for capturing packets

```
tcpdump -i eth0 -s 0 -w dumpfile
```

Wireshark Basics (cont)

- Load from file
  - Can load saved capture files generated by Wireshark or tcpdump
    - Any file in pcap save format

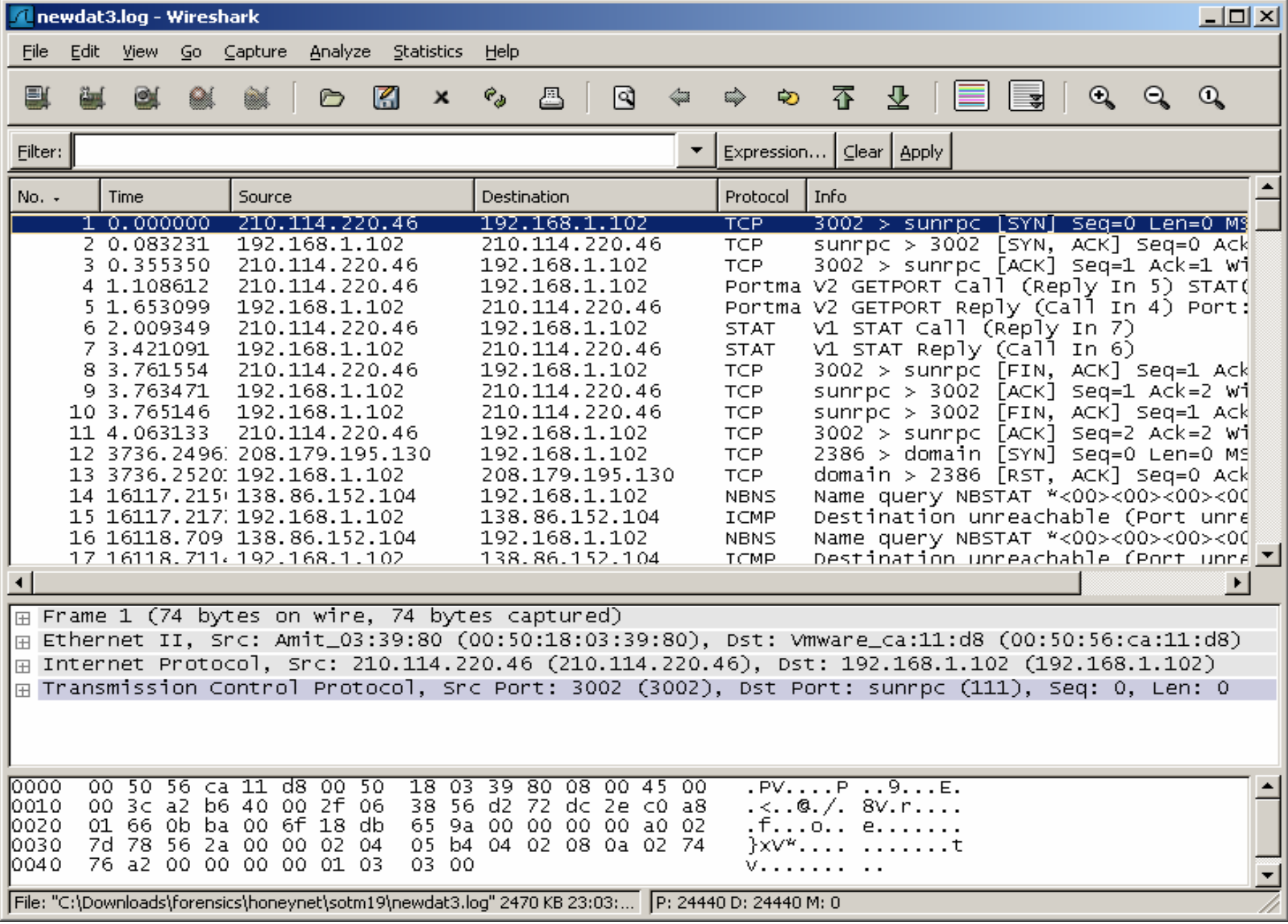
Wireshark Info



Files load faster if you disable name resolution (off by default)

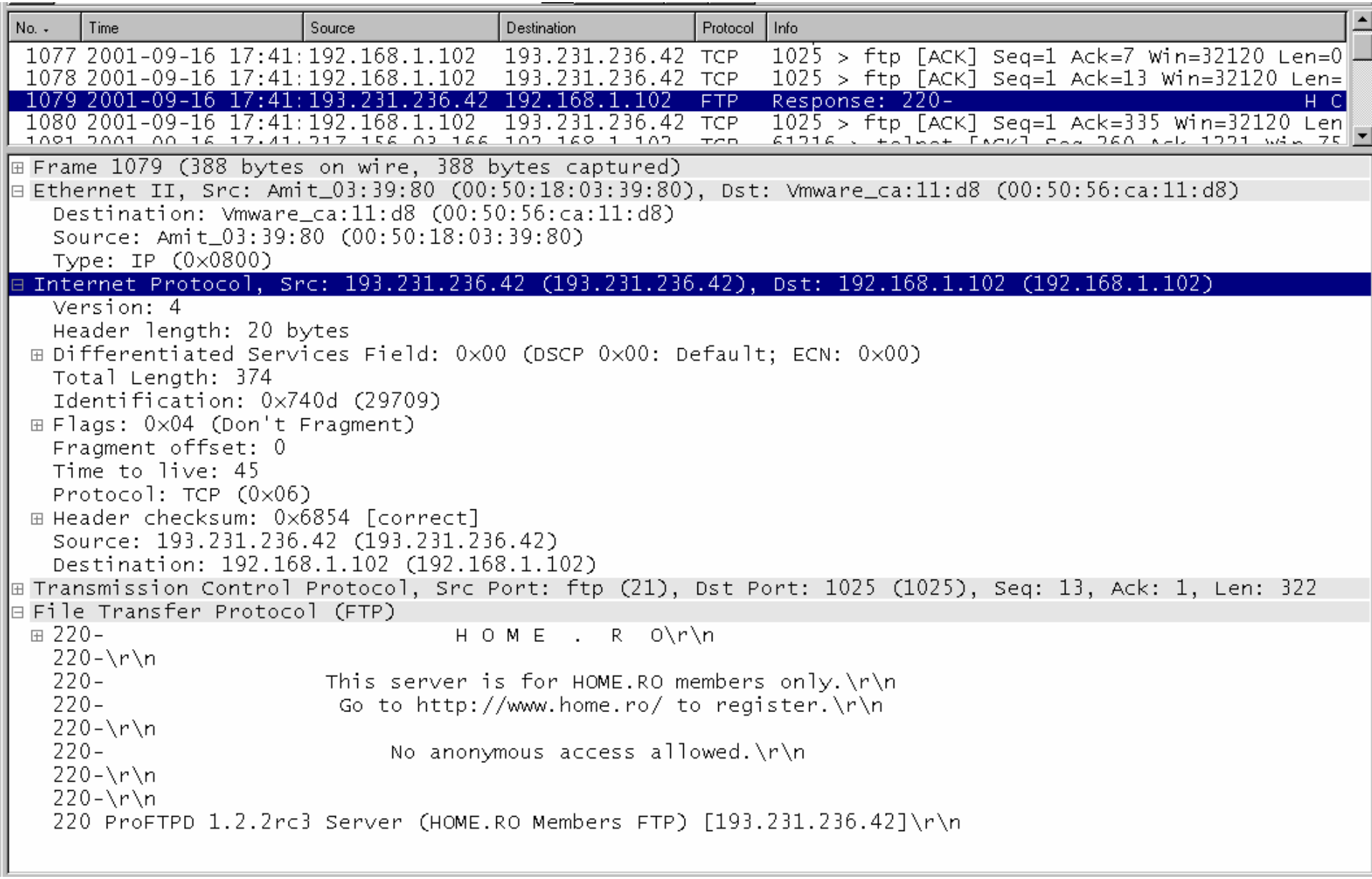
Wireshark Basics (cont)

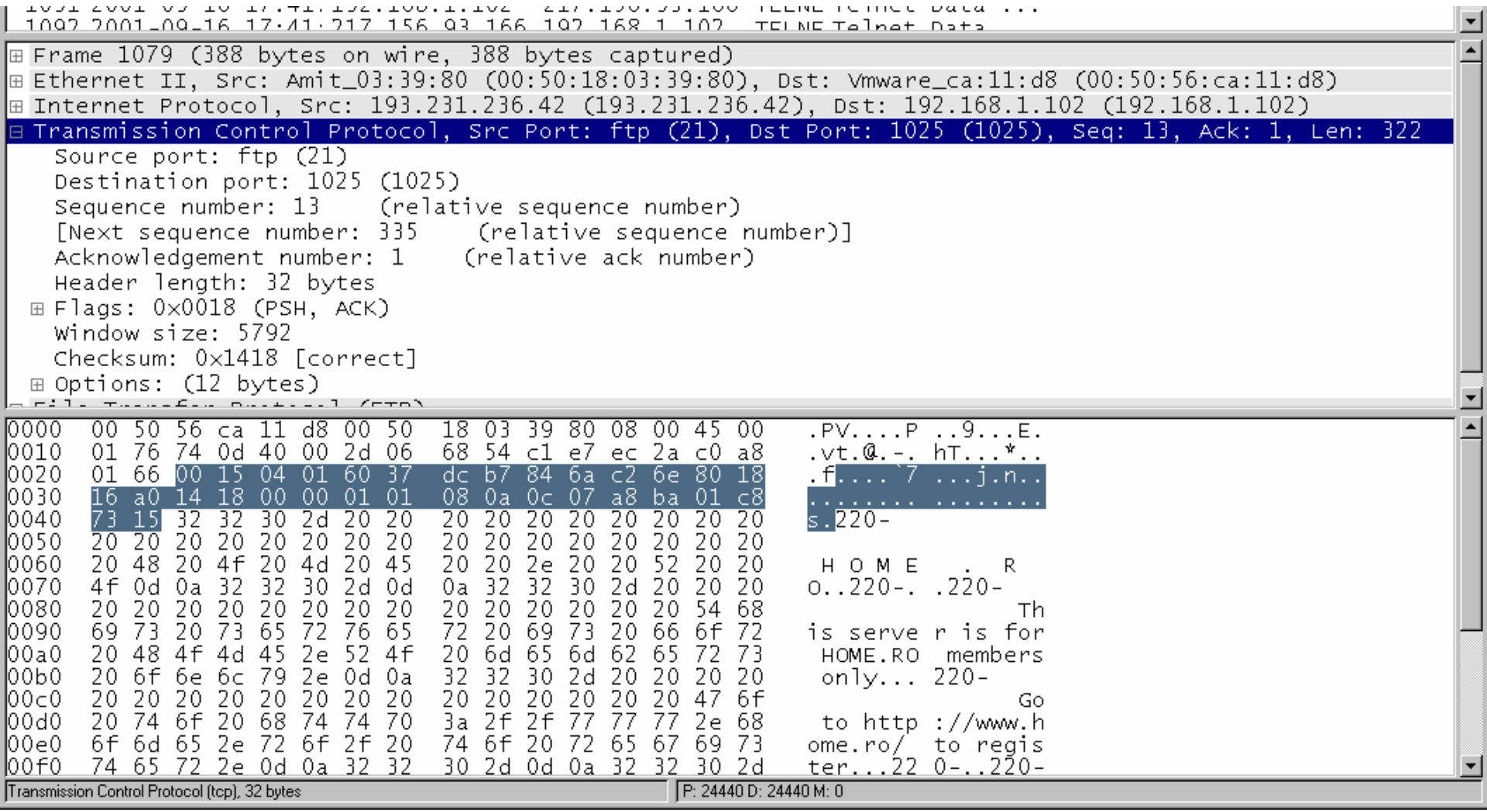
- Basic Display
  - Packet number
  - time
  - Source/Dest IP
    - Can resolve names if you choose
  - Protocol
  - Info


newdat3.log - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: Expression... Clear Apply
No. Time Source Destination Protocol Info
1 0.000000 210.114.220.46 192.168.1.102 TCP 3002 > sunrpc [SYN] Seq=0 Len=0 MS
2 0.083231 192.168.1.102 210.114.220.46 TCP sunrpc > 3002 [SYN, ACK] Seq=0 Ack
3 0.355350 210.114.220.46 192.168.1.102 TCP 3002 > sunrpc [ACK] Seq=1 Ack=1 Wi
4 1.108612 210.114.220.46 192.168.1.102 Portma V2 GETPORT Call (Reply In 5) STAT(
5 1.653099 192.168.1.102 210.114.220.46 Portma V2 GETPORT Reply (Call In 4) Port:
6 2.009349 210.114.220.46 192.168.1.102 STAT V1 STAT Call (Reply In 7)
7 3.421091 192.168.1.102 210.114.220.46 STAT V1 STAT Reply (Call In 6)
8 3.761554 210.114.220.46 192.168.1.102 TCP 3002 > sunrpc [FIN, ACK] Seq=1 Ack
9 3.763471 192.168.1.102 210.114.220.46 TCP sunrpc > 3002 [ACK] Seq=1 Ack=2 Wi
10 3.765146 192.168.1.102 210.114.220.46 TCP sunrpc > 3002 [FIN, ACK] Seq=1 Ack
11 4.063133 210.114.220.46 192.168.1.102 TCP 3002 > sunrpc [ACK] Seq=2 Ack=2 Wi
12 3736.2496 208.179.195.130 192.168.1.102 TCP 2386 > domain [SYN] Seq=0 Len=0 MS
13 3736.2520 192.168.1.102 208.179.195.130 TCP domain > 2386 [RST, ACK] Seq=0 Ack
14 16117.215 138.86.152.104 192.168.1.102 NBNS Name query NBSTAT *<00><00><00><00
15 16117.217 192.168.1.102 138.86.152.104 ICMP Destination unreachable (Port unre
16 16118.709 138.86.152.104 192.168.1.102 NBNS Name query NBSTAT *<00><00><00><00
17 16118.711 192.168.1.102 138.86.152.104 ICMP Destination unreachable (Port unre
Frame 1 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: Amit_03:39:80 (00:50:18:03:39:80), Dst: Vmware_ca:11:d8 (00:50:56:ca:11:d8)
Internet Protocol, Src: 210.114.220.46 (210.114.220.46), Dst: 192.168.1.102 (192.168.1.102)
Transmission Control Protocol, Src Port: 3002 (3002), Dst Port: sunrpc (111), Seq: 0, Len: 0
0000 00 50 56 ca 11 d8 00 50 18 03 39 80 08 00 45 00 .PV....P ..9...E.
0010 00 3c a2 b6 40 00 2f 06 38 56 d2 72 dc 2e c0 a8 .<..@./. 8V.r....
0020 01 66 0b ba 00 6f 18 db 65 9a 00 00 00 00 a0 02 .f...o.. e.......
0030 7d 78 56 2a 00 00 02 04 05 b4 04 02 08 0a 02 74 }xV*.... ....t
0040 76 a2 00 00 00 00 01 03 03 00 v..... ..
File: "C:\Downloads\forensics\honeynet\sotm19\newdat3.log" 2470 KB 23:03:...
P: 24440 D: 24440 M: 0

Wireshark Basics (cont)

- Decoded Packet Info
  - Broken down by layer
  - Within layers broken down by fields
  - Very useful so that you do not need to have protocol cheat sheets handy all the time
- Raw Packet Data
  - As hex and ASCII


No. Time Source Destination Protocol Info
1077 2001-09-16 17:41: 192.168.1.102 193.231.236.42 TCP 1025 > ftp [ACK] Seq=1 Ack=7 Win=32120 Len=0
1078 2001-09-16 17:41: 192.168.1.102 193.231.236.42 TCP 1025 > ftp [ACK] Seq=1 Ack=13 Win=32120 Len=
1079 2001-09-16 17:41: 193.231.236.42 192.168.1.102 FTP Response: 220-
1080 2001-09-16 17:41: 192.168.1.102 193.231.236.42 TCP 1025 > ftp [ACK] Seq=1 Ack=335 Win=32120 Len
Frame 1079 (388 bytes on wire, 388 bytes captured)
Ethernet II, Src: Amit_03:39:80 (00:50:18:03:39:80), Dst: Vmware_ca:11:d8 (00:50:56:ca:11:d8)
Destination: Vmware_ca:11:d8 (00:50:56:ca:11:d8)
Source: Amit_03:39:80 (00:50:18:03:39:80)
Type: IP (0x0800)
Internet Protocol, Src: 193.231.236.42 (193.231.236.42), Dst: 192.168.1.102 (192.168.1.102)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
Total Length: 374
Identification: 0x740d (29709)
Flags: 0x04 (Don't Fragment)
Fragment offset: 0
Time to live: 45
Protocol: TCP (0x06)
Header checksum: 0x6854 [correct]
Source: 193.231.236.42 (193.231.236.42)
Destination: 192.168.1.102 (192.168.1.102)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1025 (1025), Seq: 13, Ack: 1, Len: 322
File Transfer Protocol (FTP)
220- H O M E . R O\r\n
220-\r\n
220- This server is for HOME.RO members only.\r\n
220- Go to http://www.home.ro/ to register.\r\n
220-\r\n
220- No anonymous access allowed.\r\n
220-\r\n
220-\r\n
220 ProFTPD 1.2.2rc3 Server (HOME.RO Members FTP) [193.231.236.42]\r\n


1092 2001-09-16 17:41:217.156.93.166 192.168.1.102 TELNET Telnet Data
Frame 1079 (388 bytes on wire, 388 bytes captured)
Ethernet II, Src: Amit_03:39:80 (00:50:18:03:39:80), Dst: Vmware_ca:11:d8 (00:50:56:ca:11:d8)
Internet Protocol, Src: 193.231.236.42 (193.231.236.42), Dst: 192.168.1.102 (192.168.1.102)
Transmission Control Protocol, Src Port: ftp (21), Dst Port: 1025 (1025), Seq: 13, Ack: 1, Len: 322
Source port: ftp (21)
Destination port: 1025 (1025)
Sequence number: 13 (relative sequence number)
[Next sequence number: 335 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 32 bytes
Flags: 0x0018 (PSH, ACK)
Window size: 5792
Checksum: 0x1418 [correct]
Options: (12 bytes)
0000 00 50 56 ca 11 d8 00 50 18 03 39 80 08 00 45 00 .PV....P ..9...E.
0010 01 76 74 0d 40 00 2d 06 68 54 c1 e7 ec 2a c0 a8 .vt.@.-. hT...*..
0020 01 66 00 15 04 01 60 37 dc b7 84 6a c2 6e 80 18 .f....`7 ...j.n..
0030 16 a0 14 18 00 00 01 01 08 0a 0c 07 a8 ba 01 c8 ....
0040 73 15 32 32 30 2d 20 20 20 20 20 20 20 20 20 20 s.220-
0050 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0060 20 48 20 4f 20 4d 20 45 20 20 2e 20 20 52 20 20 H O M E . R
0070 4f 0d 0a 32 32 30 2d 0d 0a 32 32 30 2d 20 20 20 O..220-. .220-
0080 20 20 20 20 20 20 20 20 20 20 20 20 20 20 54 68 Th
0090 69 73 20 73 65 72 76 65 72 20 69 73 20 66 6f 72 is serve r is for
00a0 20 48 4f 4d 45 2e 52 4f 20 6d 65 6d 62 65 72 73 HOME.RO members
00b0 20 6f 6e 6c 79 2e 0d 0a 32 32 30 2d 20 20 20 20 only... 220-
00c0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 47 6f Go
00d0 20 74 6f 20 68 74 74 70 3a 2f 2f 77 77 77 2e 68 to http ://www.h
00e0 6f 6d 65 2e 72 6f 2f 20 74 6f 20 72 65 67 69 73 ome.ro/ to regis
00f0 74 65 72 2e 0d 0a 32 32 30 2d 0d 0a 32 32 30 2d ter...22 0-..220-
Transmission Control Protocol (tcp), 32 bytes
P: 24440 D: 24440 M: 0

Wireshark Basics (cont)

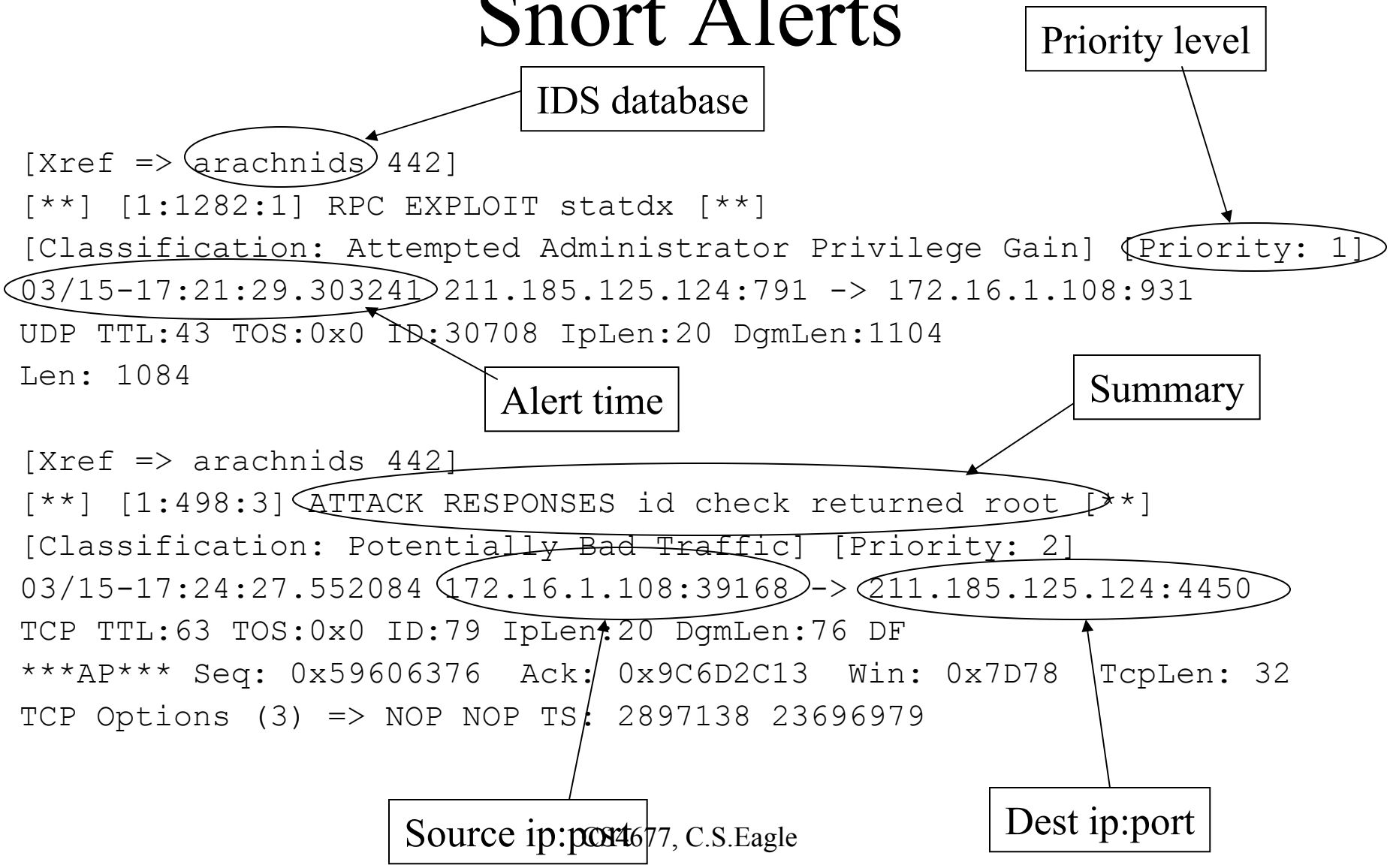
- Sorting Capability
  - Can sort display by IP, time, or Protocol
- Filtering capability
  - One of the best features
  - Capture filters use tcpdump style syntax
  - Display filters use Wireshark syntax
    - Gui expression builder available
    - Can greatly reduce amount of displayed data

```
tcp.port == 23   //telnet traffic
```

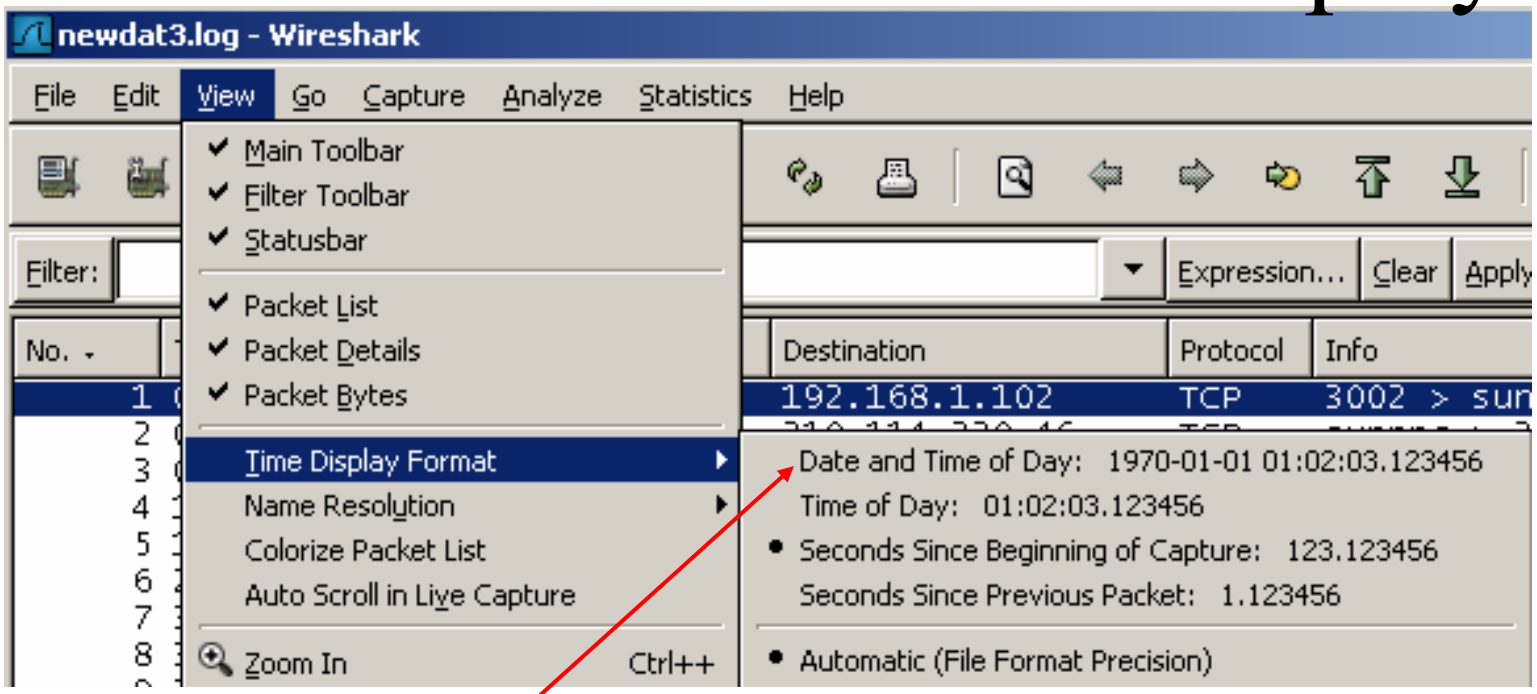
Wireshark Basics (cont)

- TCP Stream Following
  - Can rebuild an entire TCP connection
  - Display exact client/server communication sequence (data only in display window)
  - Can break out client side or server side comms separately
  - Can save data to disk for further analysis

Snort Alerts



Wireshark Time Display



Use menu to change time display format

Alert from previous snort slide

```
114 2001-03-15 17:21:24.9953 211.185.125.124
115 2001-03-15 17:21:25.0426 172.16.1.108
117 2001-03-15 17:21:25.3269 211.185.125.124
124 2001-03-15 17:21:27.3242 211.185.125.124
125 2001-03-15 17:21:29.3032 211.185.125.124
132 2001-03-15 17:21:36.3125 211.185.125.124
```

Demo: Honeynet Scan 19

- http://www.honeynet.org/scans/scan19
- Packet captures
  - Which vulnerability did the intruder exploit?
  - What ways, and in what order, did the intruder use to connect and run commands on the system?
  - How did the intruder try to hide his edits from the MAC times?
  - The intruder downloaded rootkits, what were they called?
  - Recover the rootkits from the snort binary capture
  - What does the rootkit do to hide the presence of the attacker on the system?

Re-Running Snort

- Snort can be run against a packet capture file just as easily as it can run in real time

```
snort -c /etc/snort/snort.conf -N -l . \
  -r newdat3.log
```

  - -c config file
  - -N turns off packet logging but still generates alerts
  - -l logs to the named directory
  - -r read packets from a file rather than live from the network

Interesting Alerts

```
[**] [1:1913:7] RPC STATD UDP stat mon_name format string exploit attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
09/15-19:06:07.719989 210.114.220.46:654 -> 192.168.1.102:919
UDP TTL:47 TOS:0x0 ID:41890 IpLen:20 DgmLen:1104
Len: 1076
[Xref => http://www.securityfocus.com/bid/1480]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0666]
```

36 of the following in rapid succession

```
[**] [1:1529:7] FTP SITE overflow attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
09/16-15:55:52.552709 207.35.251.172:2243 -> 192.168.1.102:21
TCP TTL:48 TOS:0x0 ID:16651 IpLen:20 DgmLen:468 DF
***AP*** Seq: 0xCF7869E4  Ack: 0xEBCD7EFE  Win: 0x7D78  TcpLen: 32
TCP Options (3) => NOP NOP TS: 237391708 29673193
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0838]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0770]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0755]
```

Interesting Alerts (cont)

```
[**] [1:1748:4] FTP command overflow attempt [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
09/16-15:55:59.485710 207.35.251.172:2243 -> 192.168.1.102:21
TCP TTL:48 TOS:0x0 ID:16786 IpLen:20 DgmLen:201 DF
***AP*** Seq: 0xCF78AE1C  Ack: 0xEBCE0EB9  Win: 0x7C70  TcpLen: 32
TCP Options (3) => NOP NOP TS: 237392403 29673724
[Xref => http://www.securityfocus.com/bid/4638]

[**] [1:498:4] ATTACK-RESPONSES id check returned root [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
09/16-15:56:01.742466 192.168.1.102:21 -> 207.35.251.172:2243
TCP TTL:64 TOS:0x10 ID:1730 IpLen:20 DgmLen:91 DF
***AP*** Seq: 0xEBCE0EB9  Ack: 0xCF78AEB5  Win: 0x7D78  TcpLen: 32
TCP Options (3) => NOP NOP TS: 29674034 237392604
```

We should probably take a look at this particular tcp connection

Wireshark Analysis

- Load the packet capture file into Wireshark
- Change the time display
- Locate the ftp session in question
- Right click on the packet and choose "Follow TCP Stream"
  - Wireshark extracts only the packets involved in this tcp connection and displays them in a separate window
  - You could save the conversation if you chose

Wireshark Analysis (cont)

- Whenever you follow a stream, Wireshark applies a filter to your data
  - Only packets that are part of the stream are displayed
  - Notice the large gap at the end of this particular conversation
    - What happened in the meantime?
    - Select last packet before the gap
    - Reset the display filter

Telnet Session

- A very revealing telnet session begins at packet 711
- Follow the stream to extract it
  - Select inbound or outbound packets to see one side of the connection or the other
- From attacker we see
  - Login as nobody followed by su to dns
  - Then initiates an ftp session

Ftp Session

- FTPs to teleport.go.ro
- User/Pass: teleport/gunoierul
- Downloads
  - Zer0.tar.gz
  - copy.tar.gz
  - ooty.tar.gz
- We can recover each of these files from the packet captures

Ftp File Recovery

- FTP is a two channel protocol
  - Command channel – port 21
    - This channel remains open for the duration of the session
  - Data channel – port 20
    - A new data stream is created for each data transfer
- Sort packets by protocol
- Filter display by port
  - tcp.port == 20

Ftp File Recovery

- Chose a packet in an FTP-DATA connection
  - Follow the stream and save
- In the main window select the last packet in the stream before resetting the filter
  - This helps you start your search for the next stream of interest

Examine File Contents

- One all of the files have been saved you can examine them
- Run *file* on them
- Extract their contents
- Determine what they do